DataHub - Knowledge-based science data management for exploratory data analysis

Thomas H. Handley, Jr.

Mark R. Rubin

Jet Propulsion Laboratory
4800 Oak Grove Drive, Pasadena, California 91109

Y. Philip Li

Aerospace Corporation
2350 East El Segundo Boulevard, El Segundo, California, 90245

ABSTRACT

It is our belief that new modes of research and new tools will be required to handle massive amount of diverse data that is to be stored, organized, accessed, distributed, visualized, and analyze. The fundamental innovation required is the integration of three automation technologies, videlicet knowledge-based expert systems, science visualization and science data management. This integration is based on a concept called the DataHub. In order to prove the concept, a series of software prototypes are being implemented. Based on user comments DataHub is continually changing. The current changes in philosophy and design are described here. In keeping with the DataHub philosophy of insulating the user from the underlying details of data formatting and computer architectures, two major changes have been made to the DataHub/X Windows interface. Additionally, because of the drawbacks of the. current implementation in label/file recognition, an expert system implementation is being investigated. Finally, short and long term design and implementation issues are discussed.

1.0 BACKGROUND AND REQUIREMENTS

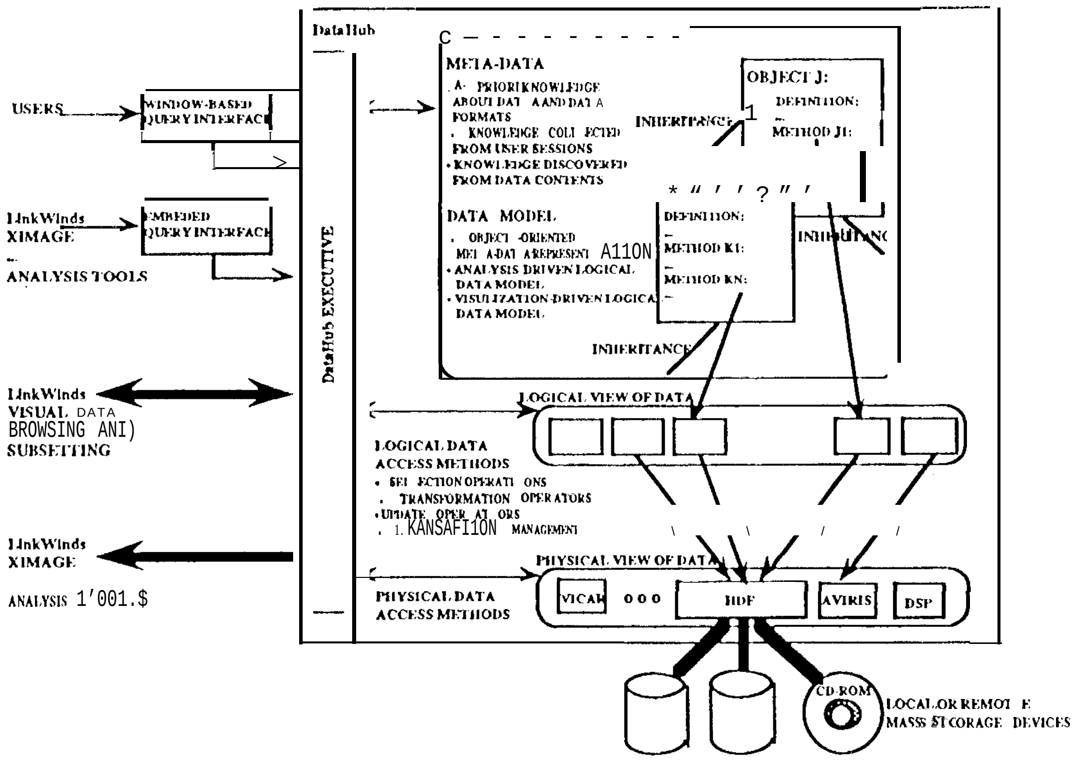
Today, after determining the existence of data, it is difficult, if not impossible, to get this data into existing tools for visualization and analysis[1]. This difficulty is generally the result of (1) incompatible data formats and the lack of available data fillers; (2) the lack of true integration between the visualization and analysis tools and the data archive system(s); (3) incompatible and/or non-c -existent metadata; and (4) the exposure of the scientist to the complexities of networking. These problems as exemplified by the current data systems will be further multiplied by the avalanche of data from future NASA missions[2, 3]. It is our belief that new modes of research and new tools will be required to handle massive amount of diverse data that is to be stored, organized, accessed, distributed, visualized, and analyzed[4, 5].

The areas of most immediate need are: (1) science data management; (2) scientific. visualization and analysis; (3) interactions in a distributed, heterogeneous environment; and (4) knowledge-basc,d assistance for these functions. The fundamental innovation required is the integration of three automation technologies, videlicet knowledge-based expert systems, science visualization and science data management. This integration is based On a concept called the DataHub.

With the DataHub, investigators will be able, to apply a complete solution to all nodes of a distributed system. Both computational nodes and interactive nodes will be able to effectively and efficiently use the data services (access, retrieval, update, etc.) in a distributed, inter- disciplinary information system in a uniform and standard way. This will enable the investigators to concentrate on their scientific endeavors, rather than to involve themselves in the intricate technical details of the. systems and tools required to accomplish their work. "Thus, investigators need not be programmers. In our initial prototypes, we are not addressing the complete networked solution, but the issues associated data management, data conversion, and data analysis within the more limited environment of networked, heterogeneous, workstations, In scientific data models, DataHub will address data driven analysis, data transformations among formats, data semantics preservation and derivation, and capture of analysis-related knowledge about the data. Expert systems are being investigated that provide intelligent assistant system(s) with some knowledge of data management and analysis built-in and eventually incorporation of mature expert system technology to aid exploratory data analysis, i.e., neural nets, classification systems. Additionally, it is a goal to investigate the capture and encoding of knowledge about the data and their associated processes. Maybe of most importance, in data analysis, the DataHub will provide data management services to exploratory data analysis applications, i.e., LinkWinds[6], an exploratory data analysis environment

2.0 DESIGN

Figure 1 depicts the functional architecture for the DataHub. The major functions of the DataHub are to provide (1) an interactive user interface; (2) a command-based query interface; (3) a set of data manipulation methods; (4) a metadata manager; and (5) an underlying science data model. The interactive user interface, basic data operators and a data interchange interface with LinkWinds have been implemented in the first prototype.



DataHub
DataHub EXECUTIVE
USERS
WINDOW-BASED QUERY INTERFACE
LinkWinds XIMAGE ... ANALYSIS TOOLS
EMBEDED QUERY INTERFACE
LinkWinds VISUAL DATA BROWSING AND SUBSETTING
LinkWinds XIMAGE ANALYSIS TOOLS
META-DATA
A-PRIORI KNOWLEDGE ABOUT DATA AND DATA FORMATS
KNOWLEDGE COLLECTED FROM USER SESSIONS
KNOWLEDGE DISCOVERED FROM DATA CONTENTS
DATA MODEL
OBJECT-ORIENTED META-DATA REPRESENTATION
ANALYSIS DRIVEN LOGICAL DATA MODEL
VISULIZATION-DRIVEN LOGICAL DATA MODEL
OBJECT J:
DEFINITION:
METHOD J1:
DEFINITION:
METHOD K1:
METHOD KN:
INHERITANCE
INHERITANCE
INHERITANCE
LOGICAL VIEW OF DATA
LOGICAL DATA ACCESS METHODS
SELECTION OPERATIONS
TRANSFORMATION OPERATORS
UPDATE OPERATORS
TRANSACTION MANAGEMENT
PHYSICAL VIEW OF DATA
PHYSICAL DATA ACCESS METHODS
VICAR
o o o
HDF
AVIRIS
DSP
CD-ROM
LOCAL OR REMOTE MASS STORAGE DEVICES

Figure 1-- Functional Architecture

Of particular note is the emerging relationship with LinkWinds. As illustrated in Figure 1, LinkWinds is providing two functions: (1) a visual data exploration or analysis environment; and (2) visual browsing and subsetting services.

The command -based query interface is designed for the data visualization system to issue data management commands to the DataHub. The data manipulation methods provide the data selection, subsetting, conversion, transformation, and updates for science data. The metadata manager captures the necessary knowledge about science data. Finally, the science data model supports the underlying object-oriented representation and across methods.

in order to prove the concept, a series of software prototypes are being implemented. The architecture of the software implementation is depicted in Figure 2. A layered architecture has been adopted for the implementation, which implies that any layer can be changed and/or replaced without affecting other layers. The top layer is the external interface that links to the human users via an interactive interface and the visualization system via a connection interface. The data model is implemented in the intelligent data management layer. The physical data access functions are built in the data interface layer.

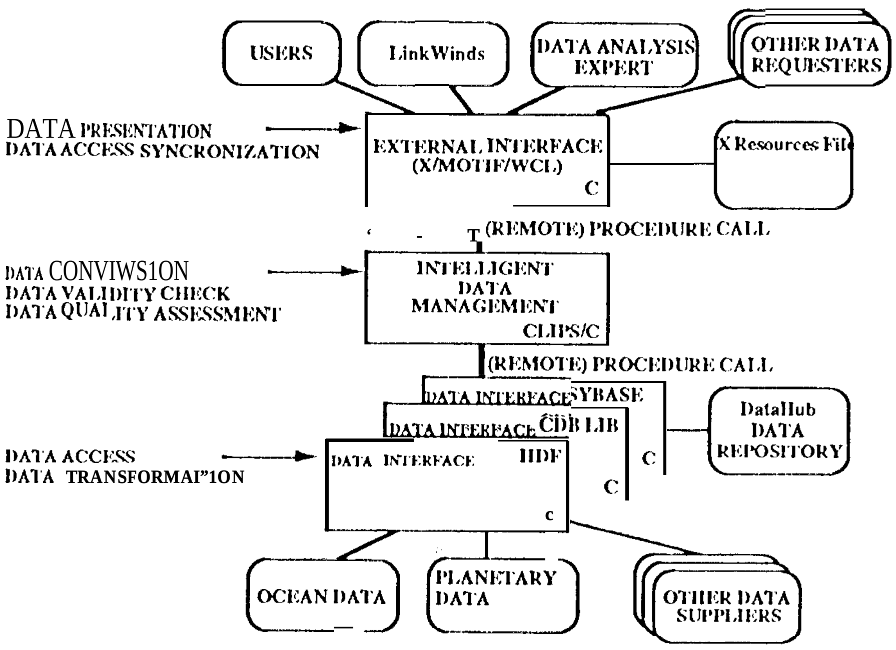


Figure 2-- Software Architecture

3.0 IMPLEMENTATION UPDATE

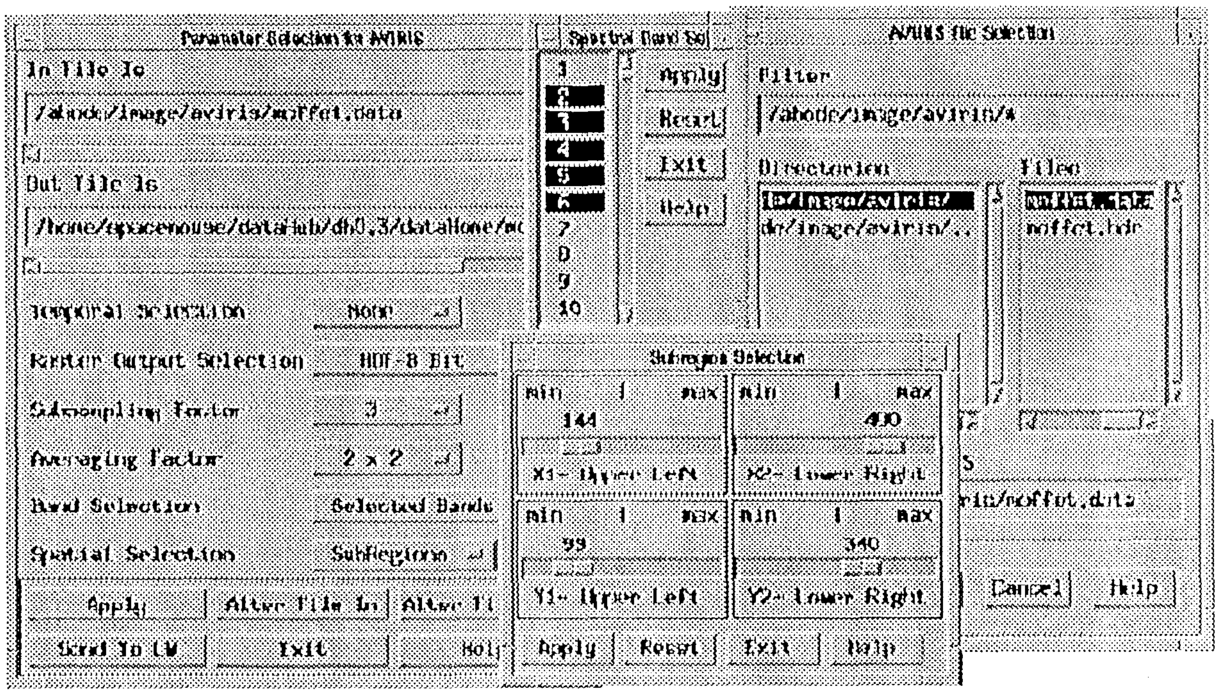
3.1 USER INTERFACE AND INTERACTIONS CHANGES

in keeping with the DataHub philosophy of insulating the user from the underlying details of data formatting and computer architectures, two major changes were made to the DataHub/X Windows interface. The first change deals with the user's graphical interface to DataHub, and the second increases the user's ability to enhance and modify DataHub's environment without requiring changes to DataHub source code.

Previous versions of DataHub, e.g., versions prior to 0.5, required the user to interact with the program in a procedural manner. This interface, as depicted in Figure 3 provided the necessary control of DataHub, and in some ways achieved its goal of "hand-holding" a beginning user through a necessarily complex sequence of parameter choices. The user would, in sequential order:

(1) Specify an input data type.
(2) Select an input data file from a set of files with the specified data type.

(3) Select an output file name.
(4) Specify parameters for the data conversion.
(5) Initiate the conversion.


Parameter Selection for AVIRIS

In File Is
/abode/image/aviris/moffet.data

Out File Is
/home/spacehouse/dataHub/dh0.3/dataHome/mo

Temporal Selection None

Raster Output Selection HDF 8 Bit

Subsampling Factor 3

Averaging Factor 2 x 2

Band Selection Selected Bands

Spatial Selection SubRegions

Figure 3 -- Previous User Interface

Though facilities existed for modifying previously made choices, this interface constrained the experienced user to a fairly rigid interaction with the DataHub. ironically, it also proved difficult for some beginning users because it presented a terse facade upon program invocation.

Keeping in mind comments received from groups of users, the interface was re-designed. This new design was implemented according to a non-sequential, non-procedural philosophy. The user can now set up the details of data conversion in the any order desired, and initiate the conversion when ready. The basic interface appears in Figure 4.

Several things are immediately apparent. First, this interface presents the user with all top-level choices immediately. A novice user can browse the menu, visually locating items that relate to his expectatio ns of the program. Second, although the user is free 10 select any menu item at any time, the overall layout of the interface suggests (but does not mandate) a canonical sequence of interactions. This sequence follows an arc suggested by human factors research, from the upper left corner of the screen, across and down 10 the lower right. Less frequently used menu items are placed out of the way in the lower left corner. Interestingly, as will be described in the discussion of X resource file customization below, the interface can be easily modified to present a right-o- left flow, and text translated into other languages, for users from non-Western cultures.

Two important face.m of this design change to DataHub are less obvious. The first concerns a side.-bcnc,fi[of DataHub's use of the industry- standard X Window System and Motif widget set. Despite the fact that this new inter face is radically different from the previous row., only a fairly small percentage of the DataHub code was re-written to achieve these changes. This is due to the fact that the X "callback" programming model of message-passing largely insulates underlying code from the user interface that invokes the code.

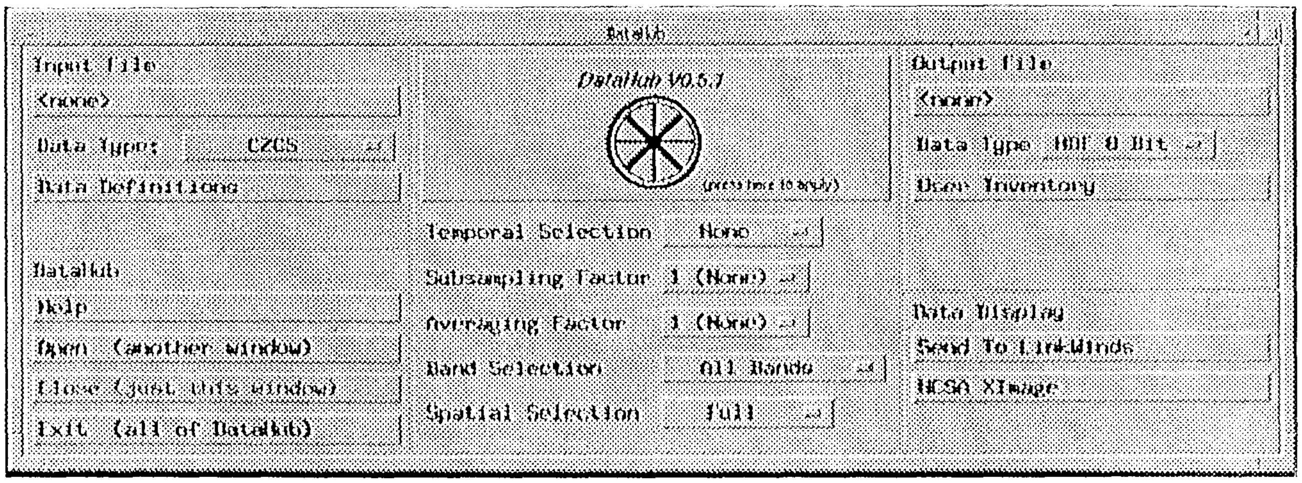


Figure 4 -- Current DataHub Interface

Additionally, it is very difficult 10 see from static pictures of the interface some. of the "behind-the-scenes" work that the program does in order to present an easy to use interface. Consider the following typical interaction with DataHub: The user first selects a desired input data type as illustrated by Figure 5.

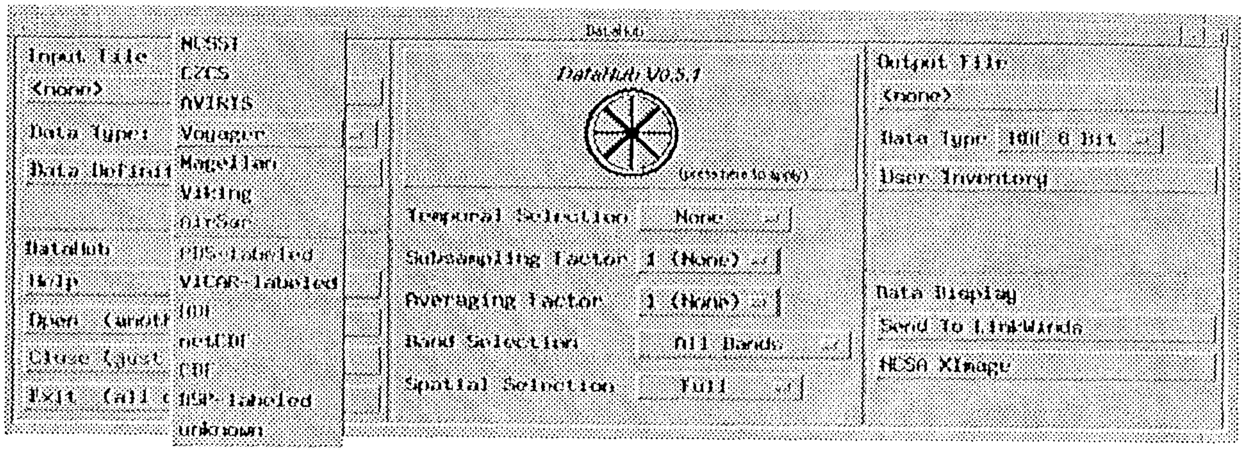
When the use.] clicks on the "Input File" name area, the user is presented with a directory listing of files of this type as depicted by Figure 6

Note that tables mapping data types 10 desired directories arc, user-customizable, as described below. However, DataHub dots not force the user into any correspondence between the input file. and directory, and the data type. }Ic.cause the Motif "File Selection Box" allows traversal to other directories, and any kind of data file. may have been inadvertently stored in any directory, DataHub (toes a preliminary scan of the file. when the. user accepts a file name. with the. "OK" button. If knowledge-based analysis 1001s in DataHub determine the file is of a different type than specified, the user is informed of this fact by an unobtrusive change of the Data Type button. If the file type is of indeterminate, the. user is notified of this as an error condition.

Although not fully implemented, see section 3.2, DATASET IDENTIFICATION EXPERT, this knowledge of the file type will cause. DataHub to upf.late all other menu areas by modify and "insensitizing" (i.e., making the menu item not selectable by the user) operations that have incorrect semantics for the current data type. "1'elm])" menus to specify spatial, temporal, and band subsetting of the data, and subsample and averaging operations, may be selected as desired and appropriate.

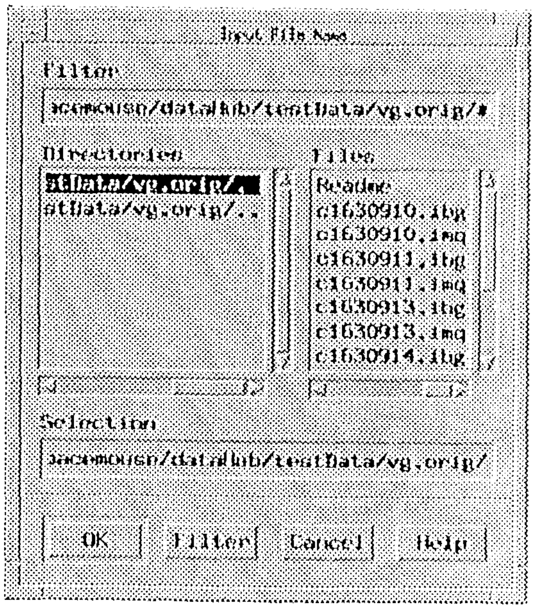
Output file selection is provided with similar context-sensitive guidance. The user may change the parameters of the conversion, convert the input data, browse the results of previous conversions, or send the data to display programs such as LinkWinds.

DataHub also allows multiple conversions to be specified simultaneously. An option in the lower-left corner of the interface allows the "cloning" of another DataHub interface. In this way, multiple interactions with similar parameterizations may appear on the screen at the same time, allowing easy cross- reference.



Temporal Selection None
Subsampling Factor 1 (None)
Averaging Factor 1 (None)
Band Selection All Bands
Spatial Selection Full

Figure 5 -- Data Type Menu



Filter: acemousn/datahub/testData/vg.orig/*

Directories:
atData/vg.orig/.
atData/vg.orig/..

Files:
Readme
c1630910.lbg
c1630910.imq
c1630911.lbg
c1630911.imq
c1630913.lbg
c1630913.imq
c1630914.lbg

Selection: acemousn/datahub/testData/vg.orig/

Figure 6 -- File Selection Box

Although not strictly mandatory, most X Windows applications require. a "resource file" to store user preferences for the appearance and use of the program. A negative side-cffcc[of this is that it complicates the installation and administration of the program, but these effects are minor, manageable, and far outweighed by the benefits this method provides.

As discussed previously, large portions of the DataHub interface arc specified in the resource file, and the layout and appearance of the program can be radically changed without modifying the underlying source code. Colors, text, and geometric positioning of the elements of the interface can all be changed.

These capabilities of the X Window System and the Motif widget set arc built on top of a general purpose database engine called the X Resource Manager[7, 8, 9, 10] Given the existence of this capability, the decision was made to store other DataHub configuration data in the X Resource file.

Note that, in many ways, this is an interim solution. Previous versions of DataHub had this kind of data hard-coded into the application itself, and any changes required the services of the DataHub development team. By the same token, future enhancements to DataHub will provide a graphical interface to these configuration and data definition facilities, providing users with no computer experience at all access to the capabilities.

The current version of DataHub store.s information such defaults for conversions of the various datatypes in the resource file. The resource file is a plain, ASCII-text file, and may be modified with any editor program of the user's choice. For instance, a section of the resource file describes parameters for the conversion of MCSST data files. This portion of the resource file is illustrated in Table 1. Other data types arc described similarly; all may be modified by the user,

*MCSST.typeLabel:	MCSST
*MCSST.datatype:	DSP_ MCSST
*MCSST.directory:	../testData/mcsst.orig
*MCSST.xMax:	2047
*MCSST.xMin:	0
*MCSST.yMax:	10?.3
*MCSST.yMin:	0
*MCSST.x 1:	0
*MCSST.y1:	0
*MCSST.x2:	2047
*MCSST.y2:	102.3
*MCSST.numOfBands:	0
* MCSST.temporalSelectionSelected:	True
*MCSST.numOfTemporalIntervals:	1
* MCSST.numOfTemporalIntervalsSelected:	1
*MCSST.numOfYears:	12
*MCSST.numOfYearsSelected:	0

Table 1-- MCSST Data Type Resource Entry

A more interesting example is that of region definitions. Users have noted that they frequently look at specific spatial subsets of their data, and have requested easy access to these areas. 'Ibis is specified through the DataHub "Spatial Select ion" menu, Figure 7

The user can specify a mnemonic name and coordinates for any number of such regions. The section of the. resource file that describes the regions looks like Table 2

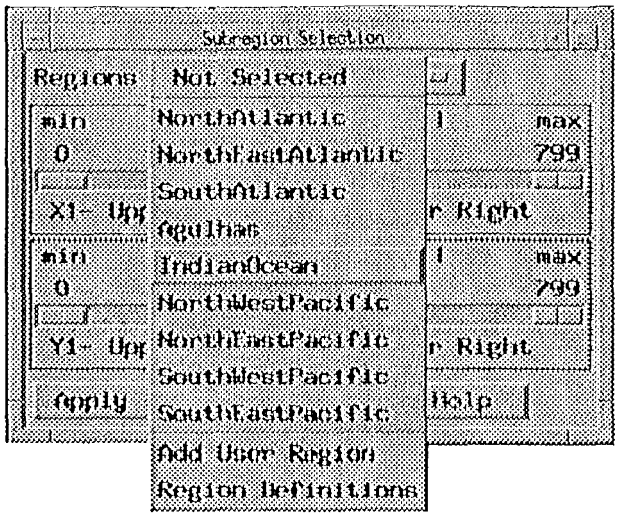


Figure 7-- Spatial Selection Popup and Region Definitions

*defaultRegions:	NorthAtlantic\			
	NorthEastAtlantic\			
	SouthAtlantic\			
	Agulhas\			
	IndianOcean\			
	NorthWestPacifi c\			
	NorthEastPacific\			
	SouthWestPacifi\			
	SouthEastPacific			
! Region	121 Begin	Lat End	Long Begin	Long End
* NorthAtlantic:	72.5097	-17.3144	261,8261	351.6504
*NorthEastAtlantic:	72.5097	-17.3144	313.5058	43.3300
*South Atlantic:	23.2910	-66.5332	290.6543	20.4785
*Agulhas:	23.2910	-66.5332	335.6543	65.4785
*IndianOcean:	30.322?.	-59.5019	31.9043	12.1.7285
*NorthWestPacific:	68.9941	-20.8300	11S.2'246	205.0488
*NorthEastPaci fic:	68.9941	-20.8300	171.4746	26 1.2988
*SouthWestPacific:	?.3.2910	-66.5332	111.0058	900.8300
*SouthEastPaci fic:	?.3.2910	-66.533?,	201.0058	290.8301

Table 2-- Regions Definitions Resources

3.2 DATASET IDENTIFICATION EXPERT

When a dataset is selected by a user, DataHub will try to indepei idently confirm the type of this dataset before any data conversion operation is performed. The way a dataset type is determined in the current prototype is proof by refutation. We try to prove that a dataset type is false based on some simple heuristics and go on to another type until we cannot prove it to bc.false. At that point, we determine that a dataset has to be of certain type, again using some simple heuristics to help the decision, and draw a conclusion. This approach is based on the assumption of a closed world of dataset types. If a dataset is not of type-a and type-b and type-c, it has 10 be of type-d, where type-d could be the unknown type. 10 an open world, it could be of type-c or type-for . . .

The drawback of this approach is that whenever wc have a ncw dataset type, the proof sequence in the dataset type scanning routine has 10 bc rechecked and in some instances bc redone. This is because the simple heuristics wc usc to help determining or rejecting a dataset type may sometimes fail to differentiate between an existing type and the newly added type. In order to tell the difference among all the, possible datatypes in a sequential process, both the sequence of tbc execution and the decision rules to guide the process have to bc checked and changed when a ncw type is added.

Because of the drawbacks of the current implementation, an expert system implementation is being investigated. As depicted in Figure 8 a rule-based engine will select and execute rules, that arc predefined based on ***a priori*** knowledge about different datasets. Major sources for the ***a priori*** knowledge come from existing data dictionaries built in diffc.rent science data communities, e.g., the Planetary Data System (PDS) Catalog. Since CLIPS[11] has been used for the metadata manager of DataHub, CLIPS will bc used here as the rrrlc-based engine. Therefore, the rules will bc forward-chaining rules of "conditions->actions" pairs. When the conditions arc satisfied, the actions will bc executed.

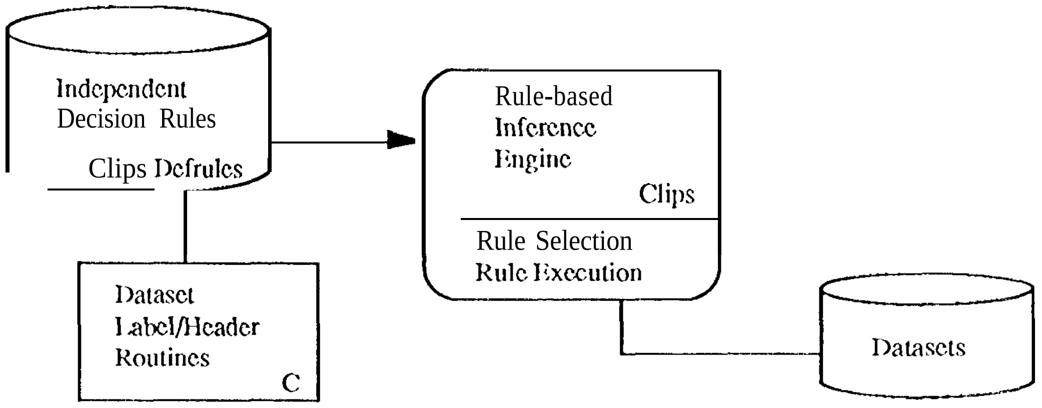


Figure 8-- Rrrlc-based Checking of Dataset Types

Within each rule, external C functions will bc invoked to check the actual dataset header to see if certain Conditions arc met. The conditions arc mainly based on the information stored in the dataset header. On the basis of the experience wc have, the dataset header can bc keyword-based ASCII text, which wc refer to as label, or binary data structures define.d by the data generator. in the current prototype, wc have knowledge about

- VICAR[12] labeled datasets, which arc used in AVIRIS, Planetary Data S ystem (PDS) Magellan, PDS Voyager, and PDS Mars Digital Image Model (MDIM) datasets,
- netCDF[13, 14] labeled datasets,
- DSP labeled datasets, which is used for Oceanography datasets published by the University of Miami,
- HDF[15] labeled datasets and
- CDF[16] labeled datasets.

The heuristics for determining the type of a dataset with ASCII label is simpler compared to the onc with binary header. For a dataset with ASCII label, wc need to examine the keywords and the corresponding value. of each keyword, and then determine if the values tells us enough about the dataset. If not, wc check more. keyword-value pairs before wc make a decision. The critical knowledge required for this decision making process is the required set of keywords and values for different dataset types. As discussed previously , the keyword/value knowledge could come from existing data dictionaries built in different science data communities. A data dictionary gives canonical description of datasets and the data items within datasets. The description of a dataset could include the structure of the dataset and the ancillary information needed to use the dataset. The data item definition include data type and the validity constraint. A data dictionary can be treated as a repository of factual knowledge about the datasets and the data items within the datasets.

The dictionaries built in different science communities carry discipline knowledge that is extremely valuable for accessing the particular kind of datasets. Therefore, the rules defined in CLIPS will also use dictionary access routines to retrieve the canonical knowledge stored in different science data dictionaries. The keyword/value pairs needed for identifying datasets need only be collected once from either the data dictionary or a human expert and be stored permanently into DataHub's persistent knowledge base, This implies that the definitions for DataHub's dataClass and dataFormat will have to be enhanced to accommodate the new data definitions assimilated.

A dataset with a binary header poses some uncertainty in the decision making process, because a binary field can be interpreted in many ways. Even if we add some validity checks to the interpretation and the binary field passes all the validity checks, there is still a chance that the binary field is just part of some other binary structure and happens to meet the constraints, An example rule for checking a dataset with binary header looks like:

```
If the dataset has a binary header,
        and the header looks like a HDF Dataset header,
        and the header does not look like a DSP Dataset header,
        and the. header does not look like a Voyager Compressed Dataset header,
Then
        The type of this dataset should be HDF.
```

The negative checks, i.e., to check that the header does not look like certain headers could be computationally intensive, for the set of condition becomes an exhaustive set and requires update when a new dataset type is added. On the other hand, if we do not use the negative checks, the positive check could be faulty because of the inherent uncertainty. This will have to be a design decision when it comes down to real implementation.

This new way of implementation should be clean and easy to maintain, One significant drawback continues in that it is fundamentally the same approach of proof by refutation with the closed world assumption, except with these two advantages. First, it allows us to update the decision heuristics easily and therefore allowing more sophisticated checking of dataset types as we learn more about each dataset. Second, the decision making process is executed in an opportunistic manner as opposed to the deterministic manner of the current implementation.

FUTURE DIRECTIONS AND WORK

The recent presentation and implementation changes were based on comments from our users. Immediate plans are to continue to enhance and expand this prototype while continuously receiving comments from the user community.

Issues for the short term include:

- DataHub provided interaction to modify and update runtime resources,
- a DataHub journal and transaction manager,
- a more general data model for user-defined data and conversion,
- expert system for dataset type checking, and
- capturing and utilizing user preferences.

Long term research and implementation issues include:

- a distributed DataHub;

- contcx[-based and content based data search capability;
- quality control and "corporate" memory of the conversion of input data points to output data points; and
- a more, general search path and browsing mechan ism,

ACKNOWLEDGMENTS

This work was supper-tcd by NASA Code SMI through Joe Bredekamp's Applied information Systems Research program. The authors wish to thank the using scientists for their valuable insights into discipline-spwific needs in data u.sage and semantics and user interactions.

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute Of Technology, under a contract with the National Aeronautics and Space Administration.

If you have comments, questions, or would like to discuss the usgs of DataHub, the authors' electronic mail addresses arc:

T. Handley	thandley@spacem ousc.jpl.nasa.gov
M, Rubin	mark@phineas.jpl .nasa.gov
P. Li	li@aero.org

REFERENCES

[1] M. Botts, informal Communication "Results of a NASA survey for EosDIS ", University of Alabama, June 2, 1992.

[2] T. Handley, and L. Preheim, "High Rate information Systems: Architectural Trends in Support of the, Interdisciplin ary Investigator", AIAA-90-5084, AIAA Second International Symposium on Space Information System, 17-19 September 1990, Pasadena, CA

[3] M. Mitchell Waddrop, "Learning to Drink from a Fire Hose", Science, Vol. 248, pp. 674-675, May 11, 1990.

[4] M. Stonebraker and J. Dozier, "Large Capacity Object Servers to Support Global Change Research". Report # 91/1, SEQUOIA 2000, University of California,

[5] Peter J. Denning, " information Technologies for Astrophysics Circa 2001- A Position Paper, NASA Workshop on Astrophysical Information Systems, May 23-25, 1990.

[6] Allen S. Jacobson and Andrew L. Berkin, "LinkWinds: A System for Interactive Scientific Data Analysis and Visualization", Proceedings Of High Performance Computing Conference, Society for Computer Simulations, March, 1 993

[7] Robert W. Scheifler and James Gett ys, X Window System, Digital Press, Burlington, MA, 1992.

[8] Paul J. Asente and Ralph R. Swick, X Window System Toolkit, Digital Press, Burlington, MA, 1992.

[9] 'J". O'Reilly, XLib Programming Manual, O'Reilly & Associates, Inc., Sebastopol, CA, 1990

[10] '1'. 0'Reilly, X Toolkit Intrinsics Programming Manual, O'Reilly & Associates, Inc., Sebastopol, CA, 1990

[11] Joseph C. Ciarrarano, CLIPS User's Guide Version 5.1, September 10 ,1991, Volumes 1-5, Johnson Space Center, information Systems Directorate, Software Technology Branch.

[12] Susan LaVoie, et. al., VICAR User's Guide Version 2, June 1, 1989, JPL D-4186, Rev A.

[13] NetCDF User's Guide Version 2.0, October 1991, Unidata Program Center, Boulder, CO.

[14] Russ Rew and Glenn Davis, "NetCDF: An Interface for Scientific Data Access", IEEE Computer Graphics & Applications, July 1990, pp. 76-82.

[15] NCSA HDF Calling Interfaces and Utilities Version 3,1, July 1990, NCSA Software Tools Group, University of Illinois at Urbana-Champaign, 11..

[16] CDF User's Guide for UNIX Systems Version 2,1, January 9, 1992, National Space Science Data Center, NASA - Goddard Space blight Center, Greenbelt, MA.